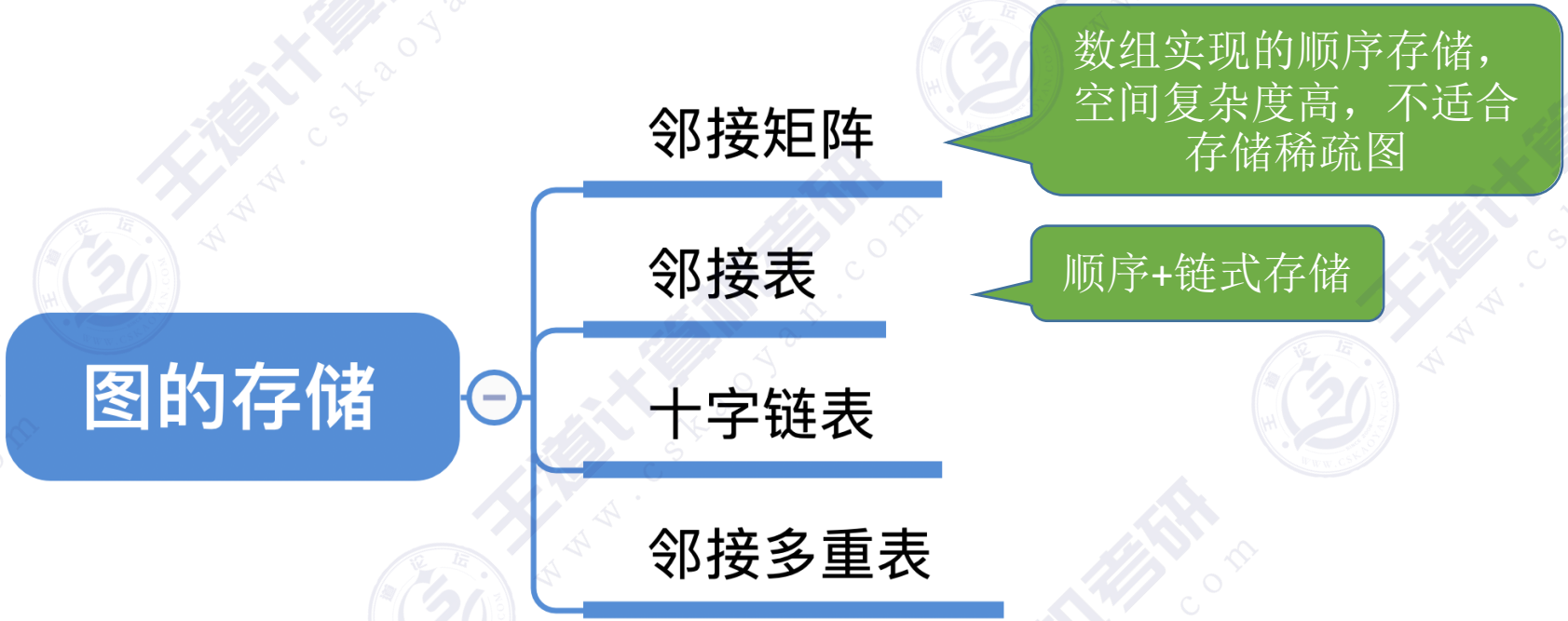
本节内容

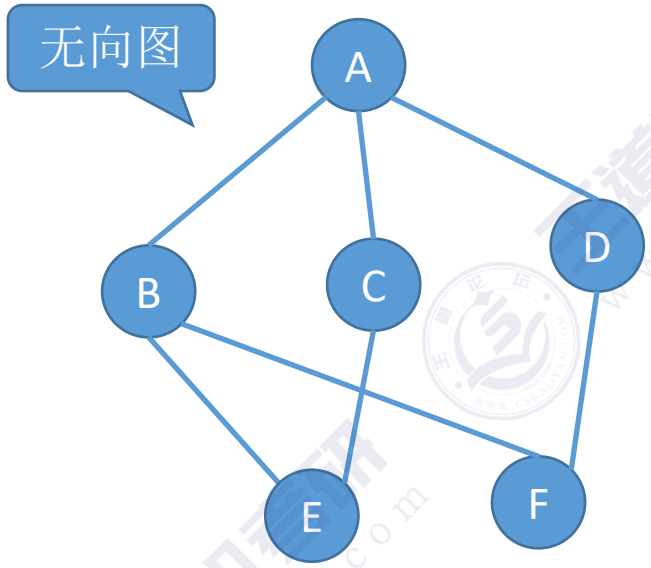
图的存储

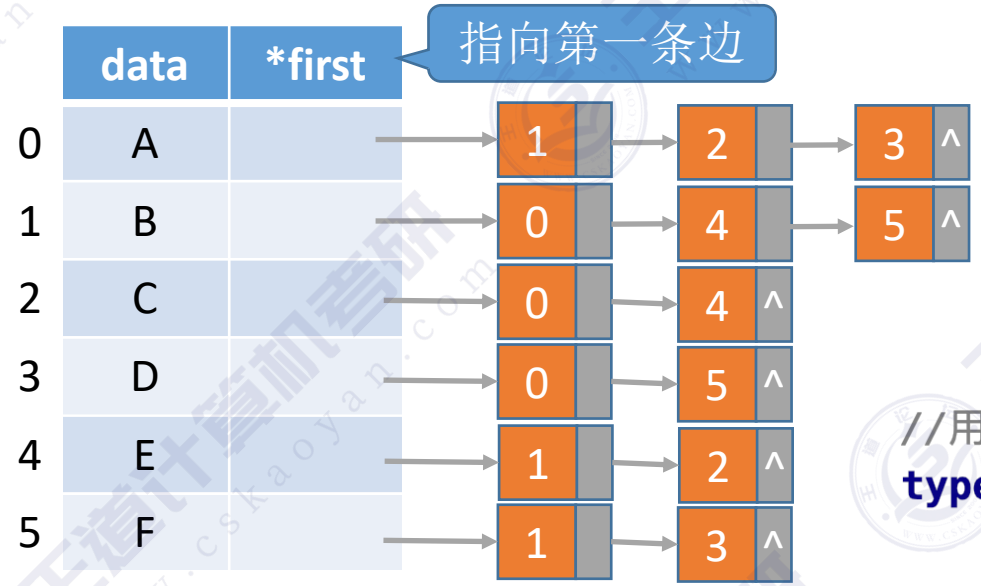
邻接表法

知识总览



邻接表法（顺序+链式存储）



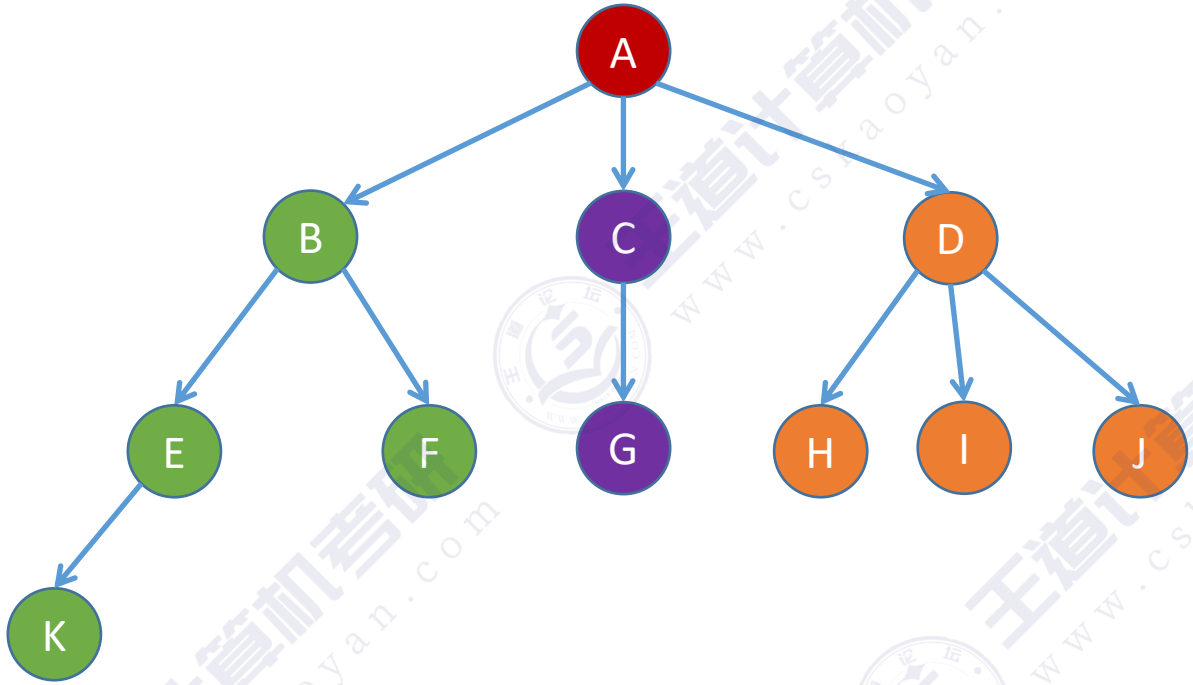



```c
//用邻接表存储的图
typedef struct{
    AdjList vertices;
    int vexnum,arcnum;
} ALGraph;
```

```c
//"边/弧"
typedef struct ArcNode{
    int adjvex;              //边/弧指向哪个结点
    struct ArcNode *next;    //指向下一条弧的指针
    //InfoType info;         //边权值
}ArcNode;
```

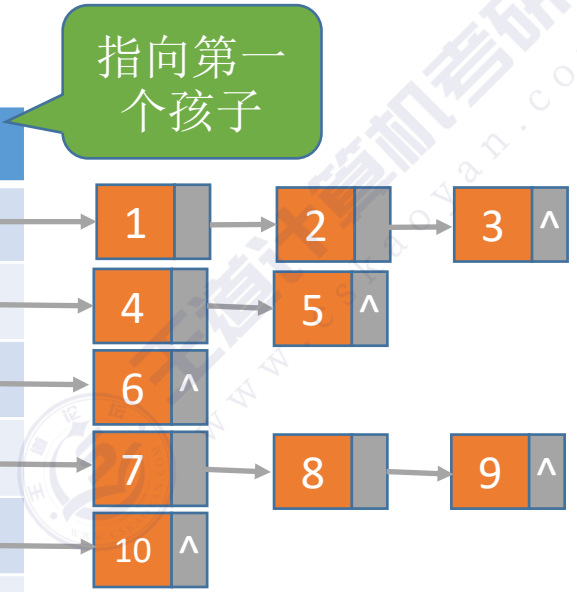
```c
//"顶点"
typedef struct VNode{
    VertexType data;     //顶点信息
    ArcNode *first;      //第一条边/弧
}VNode,AdjList[MaxVertexNum];
```

对比：树的孩子表示法

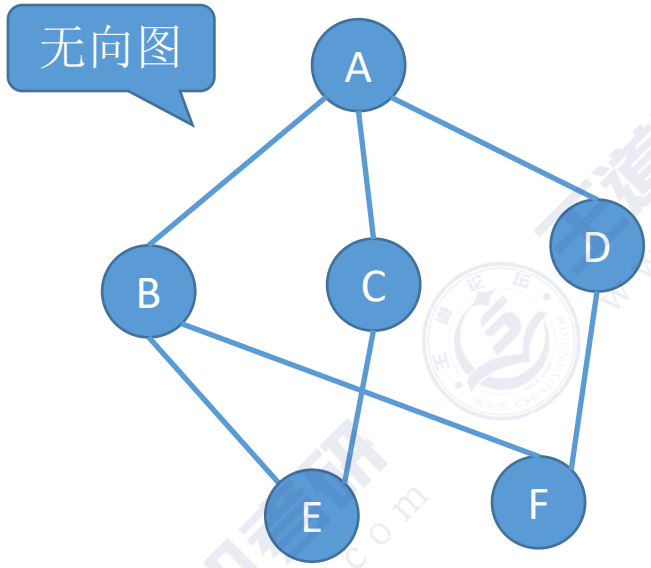


孩子表示法：顺序存储各个节点，每个结点中保存孩子链表头指针

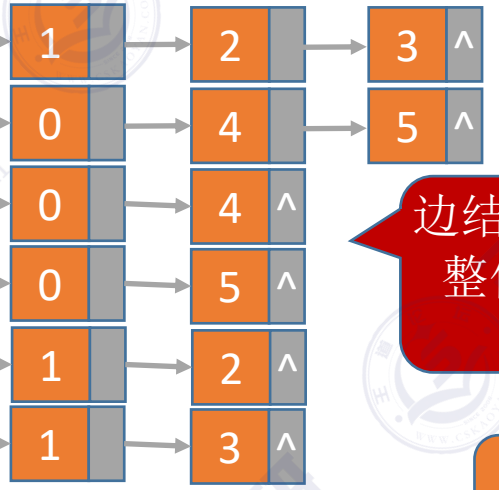
	data	*firstChild
0	A	
1	B	
2	C	
3	D	
4	E	
5	F	^
6	G	^
7	H	^
8	I	^
9	J	^
10	K	^



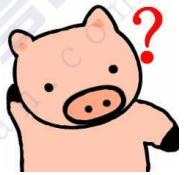
邻接表法



	data	*first
0	A	
1	B	
2	C	
3	D	
4	E	
5	F	



边结点的数量是2|E|，整体空间复杂度为O(|V| + 2|E|)

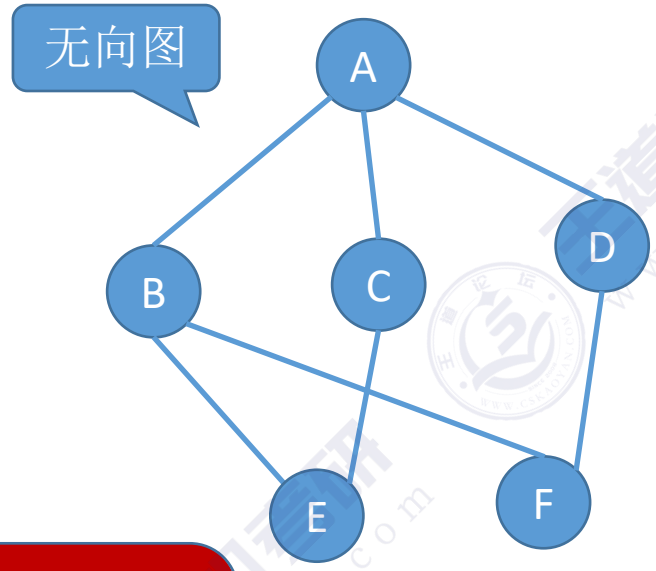


思考：如何求顶点的度、入度、出度？

如何找到与一个顶点相连的边/弧？

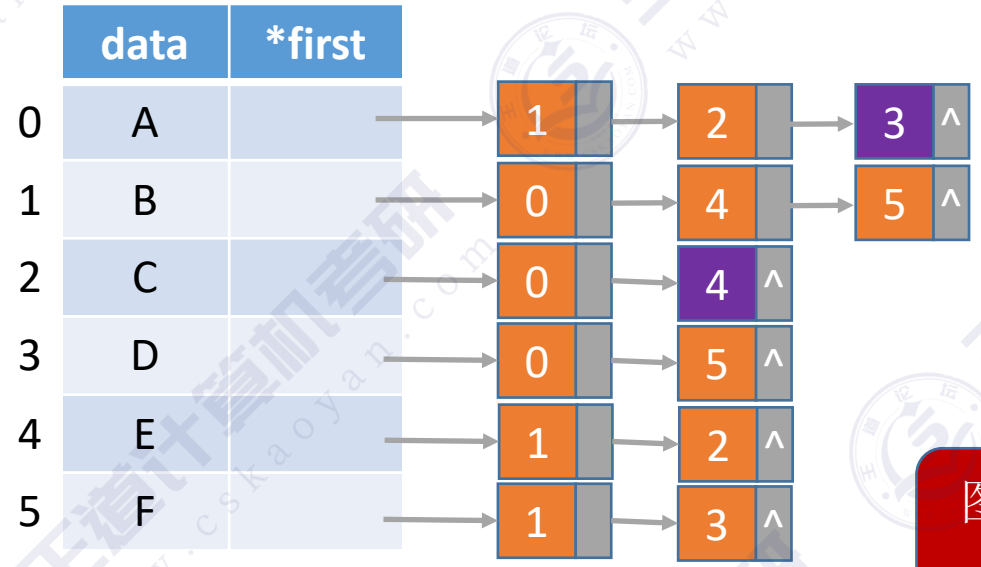
边结点的数量是|E|，整体空间复杂度为O(|V| + |E|)

邻接表法

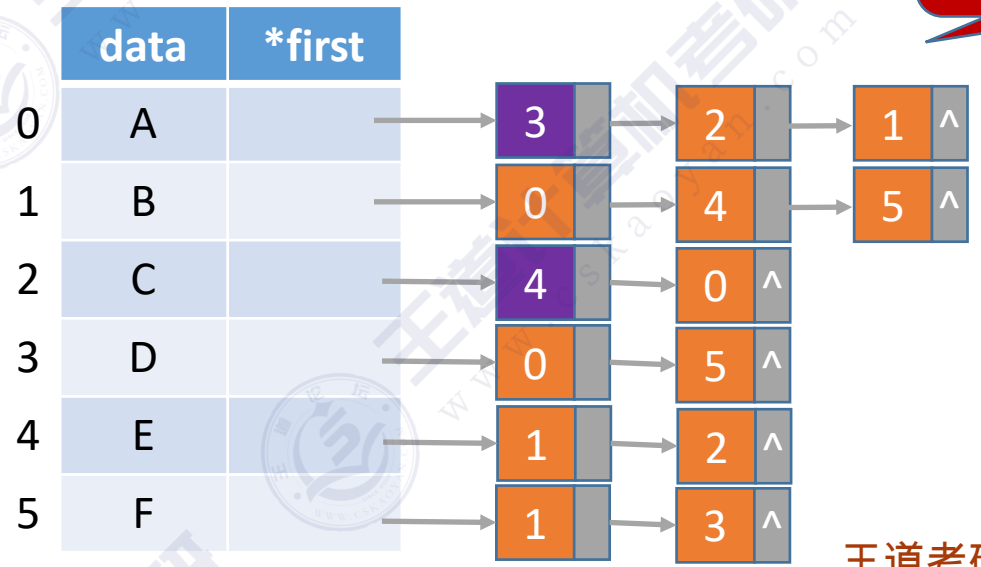


只要确定了顶点编号，图的邻接矩阵表示方式唯一

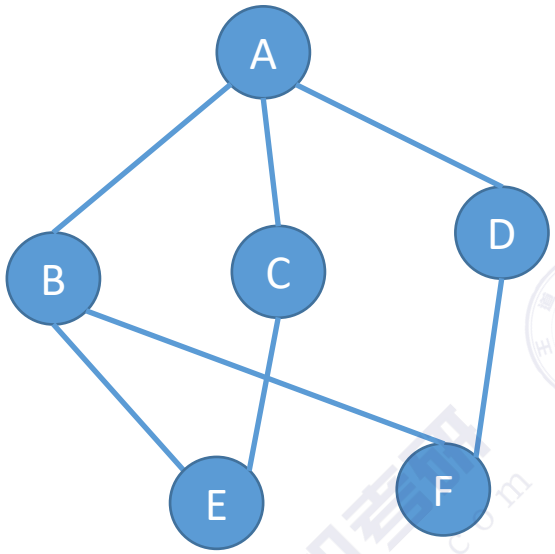
	A	B	C	D	E	F
A	0	1	1	1	0	0
B	1	0	0	0	1	1
C	1	0	0	0	1	0
D	1	0	0	0	0	1
E	0	1	1	0	0	0
F	0	1	0	1	0	0



图的邻接表表示方式并不唯一



知识回顾与重要考点



邻接表

	data	*first
0	A	→ 1 → 2 → 3 ^
1	B	→ 0 → 4 → 5 ^
2	C	→ 0 → 4 ^
3	D	→ 0 → 5 ^
4	E	→ 1 → 2 ^
5	F	→ 1 → 3 ^

邻接矩阵

	A	B	C	D	E	F
A	0	1	1	1	0	0
B	1	0	0	0	1	1
C	1	0	0	0	1	0
D	1	0	0	0	0	1
E	0	1	1	0	0	0
F	0	1	0	1	0	0

	邻接表	邻接矩阵
空间复杂度	无向图 O(|V| + 2|E|)；有向图O(|V| + |E|)	O(|V|²)
适合用于	存储稀疏图	存储稠密图
表示方式	不唯一	唯一
计算度/出度/入度	计算有向图的度、入度不方便，其余很方便	必须遍历对应行或列
找相邻的边	找有向图的入边不方便，其余很方便	必须遍历对应行或列